

# Smoothing via Adaptive Shrinkage (smash): denoising Poisson and heteroskedastic Gaussian signals

Zhengrong Xing\*

Department of Statistics, University of Chicago  
zhengrong@galton.uchicago.edu

and

Matthew Stephens

Department of Human Genetics, Department of Statistics, University of Chicago  
mstephens@uchicago.edu

May 26, 2016

## Abstract

We describe the idea of *adaptive shrinkage* (*ash*), a general purpose Empirical Bayes (EB) method for shrinkage estimation, and demonstrate its application to several signal denoising problems. The *ash* method takes as input a set of estimates and their corresponding standard errors, and outputs shrinkage estimates of the underlying quantities (“effects”). Compared with existing EB shrinkage methods, a key feature of *ash* is its use of a flexible family of *unimodal* distributions to model the distribution of the effects. The approach is not only flexible and self-tuning, but also computationally convenient because it results in a convex optimization problem that can be solved quickly and reliably. Here we demonstrate the effectiveness and convenience of *ash* by applying it to several signal denoising applications,

---

\*The authors gratefully acknowledge *please remember to list all relevant funding sources in the unblinded version*

including smoothing of Poisson and heteroskedastic Gaussian data. In both cases *ash* consistently produces estimates that are as accurate – and often more accurate – than other shrinkage methods, including both simple thresholding rules and purpose-built EB procedures. We illustrate the potential for Poisson smoothing to provide an alternative to “peak finding” algorithms for sequencing assays such as Chromatin Immunoprecipitation (ChIP-Seq). The methods are implemented in an R package, **smashr** (SMoothing by Adaptive SHrinkage in R), available from <http://www.github.com/stephenslab/smashr>.

*Keywords:* Empirical Bayes, wavelets, non-parametric regression, mean estimation, variance estimation

# 1 Introduction

Shrinkage and sparsity play a key role in many areas of modern statistics, including, for example, high-dimensional regression (Tibshirani, 1996), covariance or precision matrix estimation (Bickel and Levina, 2008), multiple testing (Efron, 2004) and signal denoising (Donoho and Johnstone, 1995, 1994). One attractive way to achieve shrinkage and sparsity is via Bayesian or Empirical Bayes (EB) methods (e.g. Clyde and George, 2000; Daniels and Kass, 2001; Efron and Tibshirani, 2002; Johnstone and Silverman, 2005). However, such methods are usually perceived to require context-specific implementations, and this overhead can limit their use in practice. Here we consider a flexible EB approach to shrinkage, which we call *adaptive shrinkage* (*ash*), whose goal is to provide a generic shrinkage method that could be useful for a range of applications. This method takes as input a vector of estimated “effects” and their corresponding standard errors, and outputs shrunk estimates of the effects (both point and interval estimates). It is fast, stable, and self-tuning, requiring no additional user input: the appropriate amount of shrinkage is learned from the input data. Here we demonstrate the flexibility, efficacy, and convenience of *ash* by applying it to several signal denoising problems in which shrinkage plays a central role.

Shrinkage methods are widely-used in signal denoising applications, because signal denoising can be accurately and conveniently achieved by shrinkage in a transformed (e.g. wavelet) domain (Donoho and Johnstone, 1994). Commonly-used shrinkage methods include both simple thresholding rules (Coifman and Donoho, 1995; Donoho and Johnstone, 1994) and EB methods (Clyde and George, 2000; Johnstone and Silverman, 2005). Being an EB method, *ash* has much in common with these previous EB methods, but generalizes them in two ways. First, *ash* allows more flexibility in the underlying distribution of effects ( $g$ ) than the Laplace or spike-and-slab distributions used in Clyde and George (2000); Johnstone and Silverman (2005); indeed, at its most general *ash* allows  $g$  to be any unimodal distribution. Second, *ash* allows for variations in precision in the transformed observations (e.g. wavelet coefficients); this plays an important role in the Poisson and heteroskedastic Gaussian settings we consider here.

In addition to demonstrating the benefits of *ash*, an important contribution of our work is to provide convenient software implementations for some commonly-encountered denoising problems. Specifically, we provide methods for smoothing Gaussian means in the presence of heteroskedastic vari-

ance, smoothing of Gaussian variances, and smoothing Poisson means. These are all settings that are relatively underserved by existing implementations. Indeed, we are unaware of any existing EB implementation for smoothing either the mean or the variance in the heteroskedastic Gaussian case. Consistent with previous studies (Antoniadis et al., 2001; Besbeas et al., 2004) we find our EB methods to be more accurate than commonly-used thresholding rules, and, in the Poisson case, competitive with a purpose-built EB method (Kolaczyk, 1999). Smoothing methods are widely used in scientific applications, and here we illustrate the potential for Poisson smoothing to provide an alternative to “peak finding” algorithms for sequencing assays such as Chromatin Immunoprecipitation (ChIP-Seq). Our methods are available in the R packages `ashr` (Adaptive SHrinkage in R) and `smashr` (SMoothing by Adaptive SHrinkage in R), available from <http://www.github.com/stephens999/ashr> and <http://www.github.com/stephenslab/smashr> respectively.

## 2 Methods

### 2.1 Adaptive shrinkage

Here we briefly outline *ash*: full details are provided in a companion paper, Stephens (2016), which applies *ash* to estimate false discovery rates in multiple testing settings. Our work here complements this, by demonstrating that *ash* provides the adaptive shrinkage estimates that its name promises.

Adaptive shrinkage is an EB method for estimating quantities  $\beta = (\beta_1, \dots, \beta_n)$  from noisy estimates  $\hat{\beta} = (\hat{\beta}_1, \dots, \hat{\beta}_n)$  and their corresponding standard errors  $\hat{s} = (\hat{s}_1, \dots, \hat{s}_n)$ . In its simplest form it assumes the hierarchical model

$$\beta_j \mid \hat{s}_j \sim g(\cdot) \tag{1}$$

$$\hat{\beta}_j \mid \beta_j, \hat{s}_j \sim N(\beta_j, \hat{s}_j^2), \tag{2}$$

where the distribution  $g$  is modelled using a mixture of zero-centered normal distributions. That is,

$$g(\cdot) = \sum_{k=0}^K \pi_k N(\cdot; 0, \sigma_k^2), \tag{3}$$

where the mixture weights  $\pi_0, \dots, \pi_K$  are non-negative and sum to 1, and  $N(\cdot; \mu, \sigma^2)$  denotes the density of a normal distribution with mean  $\mu$  and

variance  $\sigma^2$ . A key idea, which substantially simplifies inference, is to take  $\sigma_0, \dots, \sigma_K$  to be a fixed grid of values ranging from very small (e.g.  $\sigma_0 = 0$ , in which case  $g$  includes a point mass at 0) to very large. Estimating  $g$  then boils down to estimating the mixture weights  $\pi$ , which is done by maximum likelihood. Maximizing this likelihood is a convex optimization problem, and can be performed very efficiently using interior point methods (Koenker and Mizera, 2014), or more simply (though less efficiently for large problems) using an accelerated EM algorithm (Varadhan, 2014).

Given an estimate  $\hat{g}$  for  $g$ , the conditional distributions  $p(\beta_j | \hat{\beta}, \hat{s}, \hat{g})$  are analytically tractable, and the posterior mean  $E(\beta_j | \hat{\beta}, \hat{s}, \hat{g})$  provides a shrinkage point estimate for  $\beta_j$ .

The representation (3) provides a flexible family of unimodal and symmetric distributions  $g$ . Specifically, with a sufficiently large and dense grid of  $\sigma_0, \dots, \sigma_K$  the distribution  $g$  in (3) can arbitrarily accurately approximate any scale mixture of normals. This family includes, as a special case, many distributions used in shrinkage estimation, such as the “spike and slab”, double-exponential (Laplace), and  $t$  distributions (Clyde and George, 2000; Johnstone and Silverman, 2005). In this sense *ash* is more flexible than previous EB approaches. Furthermore, in many ways this more flexible approach actually *simplifies* inference, because the only parameters to be estimated are the mixture proportions (the variances are fixed).

Stephens (2016) also introduces various embellishments that are implemented in the *ashr* package, including generalizing the normal likelihood to a  $t$  likelihood, and dropping the symmetric constraint on  $g$  by replacing the mixture of normals with a more flexible (though less smooth) mixture of uniforms. We do not use these embellishments here.

## 2.2 Smoothing with Adaptive Shrinkage

We consider estimating a “spatially-structured” mean  $\boldsymbol{\mu} = (\mu_1, \dots, \mu_T)$  (and perhaps also the corresponding variance  $\boldsymbol{\sigma} = (\sigma_1^2, \dots, \sigma_T^2)$ ), from observations  $\mathbf{Y} = (Y_1, \dots, Y_T)$ , where  $t = 1, \dots, T$  indexes location in a one-dimensional space, such as time, or, as in our example later, location along the genome. By “spatially-structured” we mean that  $\mu_t$  will often be similar to  $\mu_{t'}$  for small  $|t - t'|$ , though we do not rule out occasional abrupt changes in  $\mu_t$ .

The best studied version of this problem is the homoskedastic Gaussian case; that is, the  $Y_t$  have Gaussian noise and constant variance. Here we consider the more general case of Gaussian data with spatially-structured mean

and spatially-structured (possibly non-constant) variance, and our methods provide estimates for both the mean and variance. We also consider denoising Poisson data (where the variance depends on the mean, so a spatially-structured mean implies spatially-structured variance).

In both cases we make use of “multi-scale” denoising methods, which essentially involve i) transforming the data using a multi-scale transform; ii) performing shrinkage estimation with *ash* in the transformed space; and iii) inverting the multi-scale transformation to obtain estimates in the original space. This strategy exploits the fact that a multi-scale transform will map smooth signals in the original space to sparse signals in the transformed space. Thus performing shrinkage in the transformed space induces smoothness in the original space.

For convenience we assume that  $T = 2^J$  for some integer  $J$ , as is common in multi-scale analyses.

## Gaussian data

Suppose

$$\mathbf{Y} = \boldsymbol{\mu} + \boldsymbol{\epsilon} \quad (4)$$

where  $\boldsymbol{\epsilon} \sim N_T(\mathbf{0}, D)$  with  $D$  the diagonal matrix with diagonal entries  $(\sigma_1^2, \dots, \sigma_T^2)$ . We consider, in turn, the problems of estimating  $\boldsymbol{\mu}$  when  $\boldsymbol{\sigma}$  is known; estimating  $\boldsymbol{\sigma}$  when  $\boldsymbol{\mu}$  is known; and finally the typical case of estimating  $\boldsymbol{\mu}$  and  $\boldsymbol{\sigma}$  when both are unknown.

*Estimating  $\boldsymbol{\mu}$  with  $\boldsymbol{\sigma}$  known*

We first transform the data using a multi-scale transformation, specifically a discrete wavelet transform. This involves pre-multiplying  $\mathbf{Y}$  by an orthogonal  $T \times T$  matrix  $W$  that depends on the wavelet basis chosen. Pre-multiplying (4) by  $W$  yields

$$W\mathbf{Y} = W\boldsymbol{\mu} + W\boldsymbol{\epsilon} \quad (5)$$

which we write

$$\tilde{\mathbf{Y}} = \tilde{\boldsymbol{\mu}} + \tilde{\boldsymbol{\epsilon}}, \quad (6)$$

where  $\tilde{\boldsymbol{\epsilon}} \sim N_T(0, WDW')$ .

Next we use *ash* to obtain (posterior mean) shrinkage estimates  $\hat{\tilde{\boldsymbol{\mu}}}$  for  $\tilde{\boldsymbol{\mu}}$ , which we then reverse transform to estimates for  $\boldsymbol{\mu}$ :

$$\hat{\boldsymbol{\mu}} := W^{-1} \hat{\tilde{\boldsymbol{\mu}}}. \quad (7)$$

Applying *ash* requires only point estimates for the elements of  $\tilde{\boldsymbol{\mu}}$ , and corresponding standard errors, which we obtain from the marginals of (6):

$$\tilde{Y}_j | \tilde{\mu}_j, s_j^2 \sim N(\tilde{\mu}_j, \omega_j^2), \quad (8)$$

where

$$\omega_j^2 := \sum_{t=1}^T \sigma_t^2 W_{jt}^2. \quad (9)$$

Thus to obtain  $\hat{\tilde{\boldsymbol{\mu}}}$  we apply *ash* to the estimates  $\hat{\beta}_j := \tilde{Y}_j$  with standard errors  $\hat{s}_j = \omega_j$ . (In practice it is important to group the wavelet-transformed observations  $\tilde{Y}_j$  by their resolution level before shrinking; see note below.)

In focusing on the marginals 8 we are ignoring correlations among the  $\tilde{Y}_j$ , and this is the primary simplification here. We are not alone in making this simplification; see [Silverman \(1999\)](#) for example.

The above outlines the basic strategy, but there are some important additional details:

1. Rather than use a single wavelet transform, we use the “non-decimated” (“translation invariant”) wavelet transform, which averages results over all  $T$  possible rotations of the data (effectively treating the observations as coming from a circle, rather than a line). Although not always necessary, this is a standard trick to reduce artifacts that can occur near discontinuities in the underlying signal (e.g. [Coifman and Donoho, 1995](#)).
2. The non-decimated wavelet transform yields  $T$  wavelet coefficients (transformed values of  $\mathbf{Y}$ ) at each of  $J = \log_2(T)$  resolution levels. We apply *ash* separately to the  $T$  wavelet coefficients at each resolution level, so that a different distribution  $g$  for the  $(\tilde{\mu}_j)$  is estimated for each resolution. This is the usual way that EB approaches are applied in this context (e.g. [Johnstone and Silverman, 2005](#)) and indeed is crucial because the underlying distribution  $g$  will vary with resolution (because smoothness of  $\boldsymbol{\mu}$  will vary with resolution).

3. Although we have presented the wavelet transform as a matrix multiplication, which is an  $o(T^2)$  operation, in practice both the wavelet transform and the inverse transform are implemented using efficient algorithms (Beylkin, 1992; Coifman and Donoho, 1995), implemented in the `wavethresh` package (Nason, 2013), taking only  $O(T \log_2 T)$  operations.

### *Estimating $\sigma$ with $\mu$ known*

To estimate the variance  $\sigma$  we apply wavelet shrinkage methods to the squared deviations from the mean, similar to the strategies of Delouille et al. (2004) and Cai and Wang (2008). Specifically, define

$$Z_t^2 = (Y_t - \mu_t)^2 \quad (10)$$

and note that  $\mathbb{E}(Z_t^2) = \sigma_t^2$ , so estimating  $\sigma$  is now a mean estimation problem. We tackle this using the mean estimation procedure above, effectively treating the wavelet-transformed values  $(W\mathbf{Z}^2)_t$  (where  $\mathbf{Z}^2 \equiv (Z_1^2, \dots, Z_T^2)$ ) as Gaussian when really they are linear combinations of  $\chi_1^2$  random variables. This requires an estimate of the variance of  $Z_t^2$ : we use  $\frac{2}{3}Z_t^4$ , which is an unbiased estimator of the variance. (If  $Z^2 \sim \sigma^2\chi_1^2$ , then  $\mathbb{E}(Z^4) = 3\sigma^4$ , and  $\mathbb{V}(Z^2) = 2\sigma^4$ .)

Despite the approximations made here, we have found this procedure to work well in practice in most cases, perhaps with a tendency to oversmooth quickly-varying variance functions.

### *Estimating $\mu$ and $\sigma$ jointly*

We iterate the above procedures to deal with the (typical) case where both mean and variance are unknown. To initialize we estimate the variance  $\sigma^2$  using

$$\hat{\sigma}_t^2 = \frac{1}{2} ((Y_t - Y_{t-1})^2 + (Y_t - Y_{t+1})^2) \quad (11)$$

where  $Y_0 \equiv Y_n$  and  $Y_{T+1} \equiv Y_1$  (equivalent to putting the observations on a circle).

Then we iterate:



1. Estimate  $\boldsymbol{\mu}$  as if  $\boldsymbol{\sigma}^2$  is known (with the value obtained from the previous step).
2. Estimate  $\boldsymbol{\sigma}^2$  as if  $\boldsymbol{\mu}$  is known (with the value obtained by the previous step); return to 1.

We cannot guarantee that this procedure will converge, but in our simulations we found that two iterations of steps 1-2 reliably yielded accurate results (so the full procedure consists of initialize + Steps 1-2-1-2).

### Poisson data

Consider now

$$Y_t \sim \text{Poi}(\mu_t), \quad (t = 1, \dots, T). \quad (12)$$

To estimate  $\boldsymbol{\mu}$  we apply *ash* to the Poisson multiscale models from [Kolaczyk \(1999\)](#); [Nowak and Kolaczyk \(2000\)](#); [Timmermann and Nowak \(1999\)](#), which are analogues of wavelet methods for Poisson data.

To explain, first recall the following elementary distributional result: if  $Y_1, Y_2$  are independent, with  $Y_j \sim \text{Poi}(\mu_j)$  then

$$Y_1 + Y_2 \sim \text{Poi}(\mu_1 + \mu_2) \quad (13)$$

$$Y_1 | (Y_1 + Y_2) \sim \text{Bin}(Y_1 + Y_2, \mu_1 / (\mu_1 + \mu_2)). \quad (14)$$

To extend this to  $T = 4$ , introduce the notation  $v_{i:j}$  to denote, for any vector  $v$ , the sum  $\sum_{t=i}^j v_t$ . Then

$$Y_{1:4} \sim \text{Poi}(\mu_{1:4}) \quad (15)$$

$$Y_{1:2} | Y_{1:4} \sim \text{Bin}(Y_{1:4}, \mu_{1:2} / \mu_{1:4}) \quad (16)$$

$$Y_1 | Y_{1:2} \sim \text{Bin}(Y_{1:2}, \mu_1 / \mu_{1:2}) \quad (17)$$

$$Y_3 | Y_{3:4} \sim \text{Bin}(Y_{3:4}, \mu_3 / \mu_{3:4}). \quad (18)$$

Together these models are exactly equivalent to  $Y_j \sim \text{Poi}(\mu_j)$ , and they decompose the overall distribution  $Y_1, \dots, Y_4$  into parts involving aspects of the data at increasing resolution: (15) represents the coarsest resolution (the sum of all data points), whereas (17) and (18) represent the finest resolution, with (16) in between. Further, this representation suggests a reparameterization, from  $\boldsymbol{\mu} = (\mu_1, \mu_2, \mu_3, \mu_4)$  to  $\mu_{1:4}$  plus the binomial parameters  $\boldsymbol{p} = (\mu_{1:2} / \mu_{1:4}, \mu_1 / \mu_{1:2}, \mu_3 / \mu_{3:4})$ , where  $p_1$  controls lower-resolution changes in the mean vector  $\boldsymbol{\mu}$  and  $p_2, p_3$  control higher resolution changes.

This idea extends naturally to  $T = 2^J$  for any  $J$ , reparameterizing  $\boldsymbol{\mu}$  into its sum  $\mu_{1:T}$  and a vector  $\mathbf{p}$  of  $T - 1$  binomial probabilities that capture features of  $\boldsymbol{\mu}$  at different resolutions. This can be thought of as an analogue of the Haar wavelet transform of  $\boldsymbol{\mu}$  for Poisson data.

Note that, in this reparameterization,  $p_j = 0.5 \forall j$  corresponds to the case of a constant mean vector, and values of  $p_j$  far from 0.5 correspond to large changes in  $\mu$  (at some scale). Thus estimating a spatially-structured  $\boldsymbol{\mu}$  can be achieved by shrinkage estimation of  $\mathbf{p}$ , with shrinkage towards  $p_j = 0.5$ . Both [Kolaczyk \(1999\)](#) and [Timmermann and Nowak \(1999\)](#) use purpose-built Bayesian models to achieve this shrinkage, by introducing a prior distribution on elements of  $\mathbf{p}$  that is a mixture of a point mass at 0.5 (creating shrinkage toward 0.5) and a Beta distribution. Here we take a different approach, reparameterizing  $\alpha_j = \log(p_j/(1 - p_j))$ , and then using *ash* to shrink  $\alpha_j$  towards 0.

To apply *ash* we need an estimate  $\hat{\alpha}_j$  and corresponding standard error  $\hat{s}_j$  for each  $j$ . This involves estimating a log-odds ratio, and its standard error, which is a well-studied problem (e.g. [Gart and Zweifel, 1967](#)). The main challenge is in dealing satisfactorily with cases where the maximum likelihood estimator for  $\alpha_j$  is infinite. We use estimates based on results from [Gart and Zweifel \(1967\)](#); see Appendix [B](#).

The simplest way to estimate  $\boldsymbol{\mu}$  is to estimate  $\alpha_j$  by its posterior mean, as output by *ash*, and then reverse the above reparameterization. The resulting estimate of  $\mu_t$  is the exponential of the posterior mean for  $\log(\mu_t)$  (because each  $\log(\mu_t)$  is a linear combination of  $\alpha_j$ ). Alternatively we can estimate  $\mu_t$  by approximating its posterior mean using the Delta method; see Appendix [B](#). Both methods are implemented in our software; for the results here we use the latter method to be more comparable with previous approaches that estimate  $\mu$  on the raw scale rather than log scale.

As for the Gaussian case, we actually use a translation invariant version of the above procedure; see Appendix [B.3](#)

## 3 Results

### 3.1 Illustration

Figure [1](#) illustrates the key features of *ash* in the context of smoothing a Gaussian signal. The data (panel (a)) is first transformed into wavelet coeffi-

cients (WCs) at different scales. Each such “observed” WC can be thought of as a noisy estimate of some “true” WC for the unknown mean that we wish to estimate. Each WC also has an associated standard error that depends on the variance of the data about the mean. The idea behind wavelet denoising is to “shrink” the observed WCs towards 0, which produces a smoother estimate of the mean than the observed data.

A crucial question is, of course, how much to shrink. A key idea behind *ash* is that the shrinkage is “adaptive”, in that it is determined by the data, in two distinct ways. First, if at a particular scale many observed WCs are “large” (compared with their standard errors) then *ash* infers that, at this scale, many of the true WCs are large - that is, the estimated distribution  $g$  in (1) will have a long tail. Consequently *ash* will shrink less at this scale than at scales where few observed WCs are large, for which the estimated  $g$  will have a short tail. This is illustrated in panels (b)-(c): at scale 1, many observed WCs are large (b), and so very little shrinkage is applied to these estimates (c). In contrast, at scale 7, few observed WCs are large (b), and so stronger shrinkage is applied (c). Second, because the likelihood 2 incorporates the standard error of each observation, shrinkage is adaptive to the standard error: at a given scale, WCs with larger standard error are shrunk more strongly than WCs with small standard error. This is illustrated in panel (d). (The standard errors vary among WCs because of the heteroskedastic variance.)

The end result is that i) data that are consistent with a smooth signal, get smoothed more strongly; ii) smoothing is stronger in areas of the signal with larger variance. In this example the smoothed signal from *ash* is noticeably more accurate than using TI-thresholding (with variance estimated by running median absolute deviation, RMAD; see Gao (1997)) (panel (e)).

## 3.2 Simulations

We conducted extensive simulation studies to compare the performance of our method, SMASH (SMoothing with Adaptive SHrinkage), with existing approaches. In particular, simulations with Gaussian noise were conducted within a [Dynamical Statistical Comparison](#) (DSC) framework, which was designed to facilitate comparisons among statistical methods. The code for this DSC is available at [dscr-smash](#), which also contains instructions for reproducing and extending the results presented here.

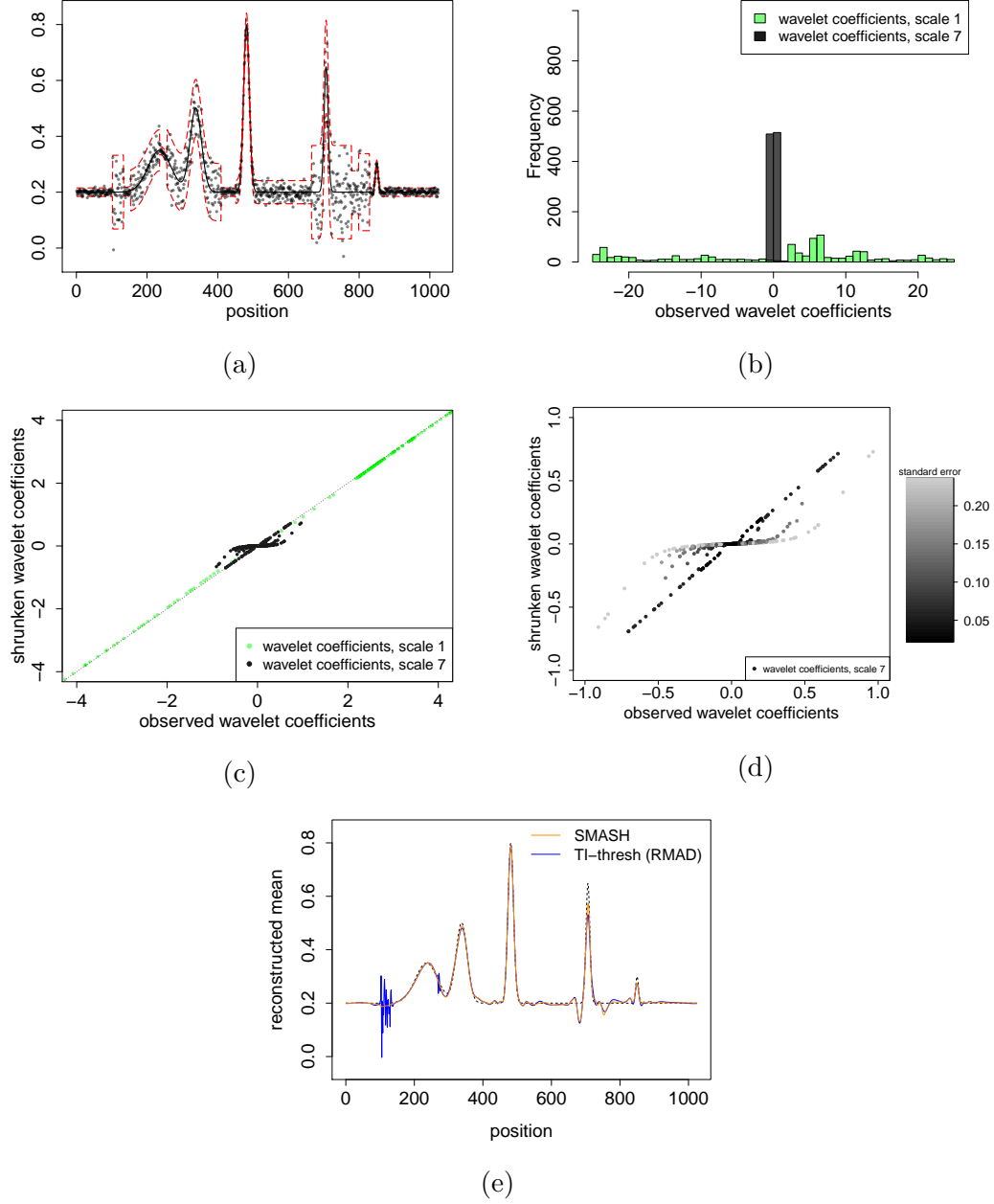


Figure 1: Illustration of both *ash* and SMASH. Panel (a) shows the “Spikes” mean function (black)  $\pm 2$  standard deviations (red), and corresponding simulated data. Panel (b) contrasts the distributions of the (true) wavelet coefficients (WCs) at two different scales - one coarse (scale 1) and one fine (scale 7). The scale 7 WCs are much more peaked around 0, because the signal is much smoother at that scale. Panel (c) contrasts the *ash* shrinkage for these two scales: the scale 7 WCs are strongly shrunk, whereas the scale 1 WCs are not. This is because *ash* infers from the data that the scale 7 WCs are peaked around 0, and consequently shrinks them more strongly. Panel (d) illustrates that *ash* shrinks WCs differently depending on their precision. Specifically WCs that are less precise are shrunk more strongly. Panel (e) plots the estimated mean functions from SMASH and TI-thresh against the true mean function; TI-thresh has some notable artifacts.

## Gaussian mean estimation

We focus initially on the homoskedastic case, modelling our simulation study after [Antoniadis et al. \(2001\)](#). Specifically we used many of the same test functions, a variety of signal lengths ( $T$ ), and two different signal to noise ratios (SNRs). We compare SMASH with Translation Invariant (TI) thresholding ([Coifman and Donoho, 1995](#)), which was one of the best-performing methods in [Antoniadis et al. \(2001\)](#), and with the Empirical Bayes shrinkage procedure Ebayesthresh ([Johnstone and Silverman, 2005](#)). All methods were applied using the Symmlet8 wavelet basis ([Daubechies, 1992](#)).

Figure 2 compares the mean integrated squared errors (MISEs) of the methods for the “Spikes” mean function, with SNR=3 and  $T=1024$ . We applied SMASH in three ways, the first (SMASH) estimating the variance function allowing for heteroskedasticity, the second estimating the variance assuming homoskedasticity (SMASH-homo) and the third using the true variance function (SMASH true variance), which could be viewed as a “gold standard”. All three versions of SMASH outperform both EbayesThresh and TI-thresholding. The different SMASH versions perform similarly, demonstrating that in this case there is little cost in allowing for heteroskedasticity when the truth is homoskedastic. We obtained similar results for other mean functions, SNRs and sample sizes (see Supplementary Materials).

Turning to heteroskedastic errors, we again compare SMASH with EbayesThresh (which assumes homoskedastic variance) and TI-thresh. For TI-thresh we considered three different ways of estimating the heteroskedastic variance: RMAD ([Gao, 1997](#)), the SMASH estimated variance, and the true variance. (TI-thresh with homoskedastic variance performed very poorly; Supplementary Results.) Figure 3 shows results for two sets of test functions: the “Spikes” mean function with the “Clipped Blocks” variance function and the “Corner” mean function with “Doppler” variance function, both with SNR=3 and  $T = 1024$ .

To summarize the main patterns in Figure 3:

1. SMASH outperforms all TI-thresh variants (including TI with the true variance).
2. SMASH performs almost as well when estimating the variance as when given the true variance.
3. Allowing for heteroskedasticity within SMASH can substantially improve accuracy of mean estimation (compare SMASH with SMASH-homo and EbayesThresh).

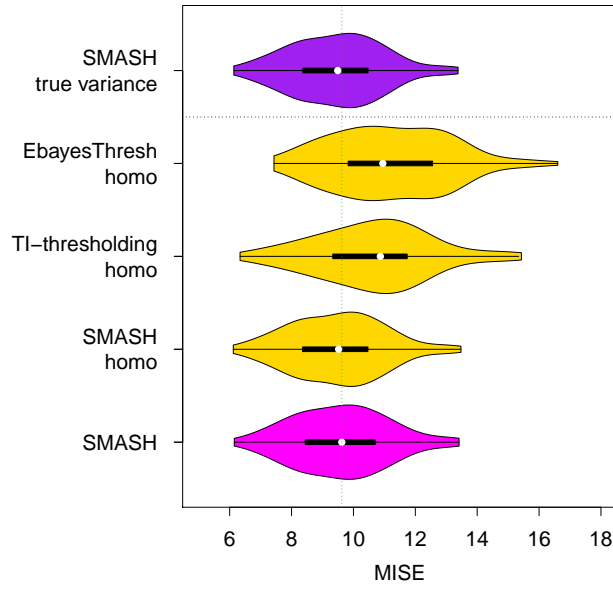


Figure 2: Comparison of accuracy (MISE) of estimated mean curves for data simulated with homoskedastic Gaussian errors. Results are for the “Spikes” mean function shown in Figure 1. The figure shows violin plots of MISEs for (from bottom to top) SMASH, SMASH with homoskedastic assumption, TI-thresholding with homoskedastic assumption, Ebayesthresh with homoskedastic assumption, and SMASH with known true variance. Colors highlight certain features of a method: yellow for methods that assume homoskedastic variance; dark purple for methods that are provided the true variance; magenta for the default SMASH method that estimates both mean and variance. Smaller MISE implies better performance; dashed green line indicates the median MISE for SMASH. SMASH outperforms both TI-thresholding and Ebayesthresh, and SMASH with estimated variance performs nearly as well as with true variance.

4. TI-thresh performs considerably better when used with the SMASH variance estimate than with the RMAD variance estimate.

These main patterns hold for a variety of different mean and variance functions, SNRs, and sample sizes (see Supplementary Materials). Some variance functions are harder to estimate than others (e.g. the “Bumps” function), and in these cases providing methods the true variance can greatly increase accuracy compared with estimating the variance. As might be expected, the gain in allowing for heteroskedastic variance tends to be greatest when the variance functions are more volatile.

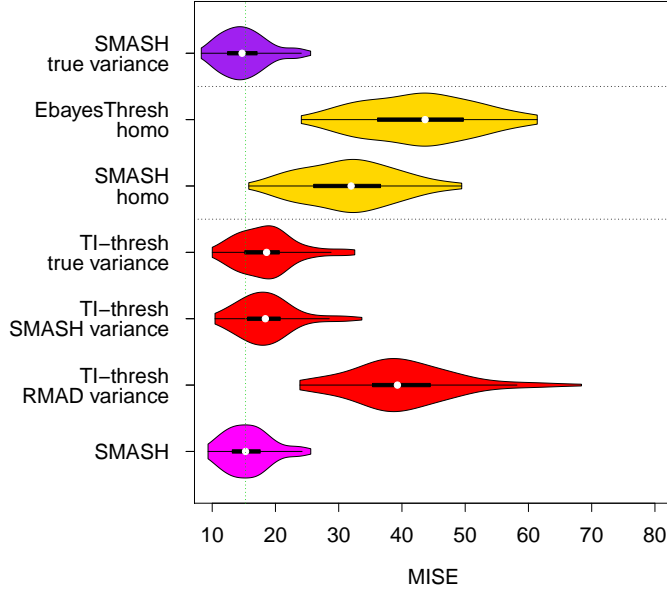
### 3.2.1 Gaussian variance estimation

One unusual feature of SMASH is that it performs joint mean and variance estimation. Indeed, we found no existing R packages for doing this. The only work we have found on wavelet-based variance estimation is [Cai and Wang \(2008\)](#), which applied a wavelet thresholding approach to first order differences. Previous related non-wavelet-based work includes [Fan and Yao \(1998\)](#), which estimates the variance by smoothing the squared residuals using local polynomial smoothing, [Brown and Levine \(2007\)](#), which uses difference-based kernel estimators, and [Menictas and Wand \(2015\)](#), which introduces a Mean Field Variational Bayes (MFVB) method for both mean and variance estimation. In no case could we find publicly-available software implementations of these methods. However, we did receive code implementing MFVB by email from M. Menictas, which we use in our comparisons here.

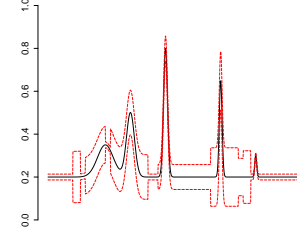
The MFVB method is based on penalized splines, and so is not well suited to many standard test functions in the wavelet literature, which often contain “spiky” local features not well captured by splines. Hence, we compared SMASH and MFVB on some smoother mean and variance (standard deviation) functions, specifically scenario A in Figure 5 from [Menictas and Wand \(2015\)](#) (Figure 4), using scripts kindly provided by M. Menictas.

We simulated data under two different scenarios:

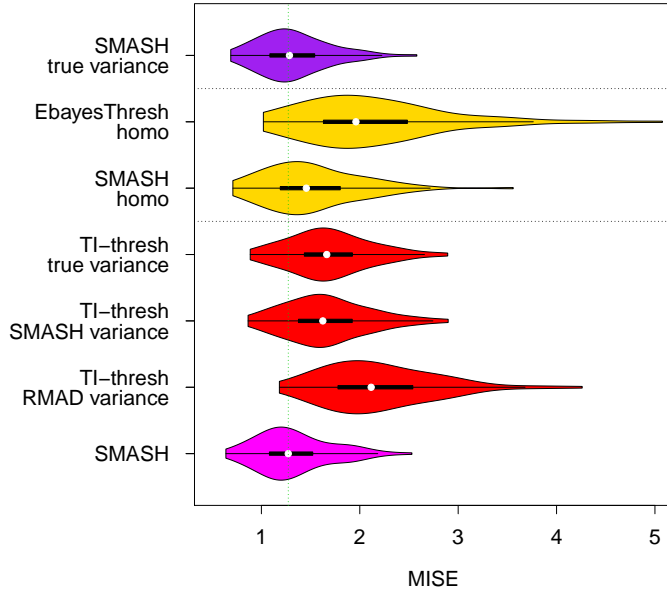
1. We generated  $T = 500$  independent  $(X_t, Y_t)$  pairs, with  $X_t \sim \text{Uniform}(0,1)$ , and  $Y_t|X_t = x_t \sim N(m(x_t), s(x_t)^2)$  where  $m(\cdot)$  and  $s(\cdot)$  denote the mean and standard deviation functions (Figure 4). We measured performance by the MSE evaluated at 201 equally spaced points on  $(X_{\min}, X_{\max})$  for both the mean and the standard deviation.



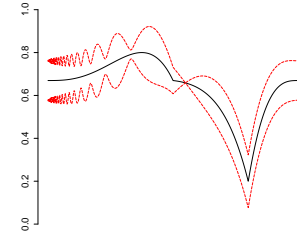
(a)



(b)



(c)



(d)

Figure 3: Comparison of accuracy (MISE) of estimated mean curves, for data simulated with heteroskedastic Gaussian errors. Panels (a) and (c) show violin plots of MISEs for various methods on two sets of mean-variance functions, shown as mean  $\pm 2$  standard deviations in panels (b) and (d). In (a) and (c) TI-based methods are shown in red, while other colors are as in Figure 2. Dashed green line indicates SMASH median MISE.



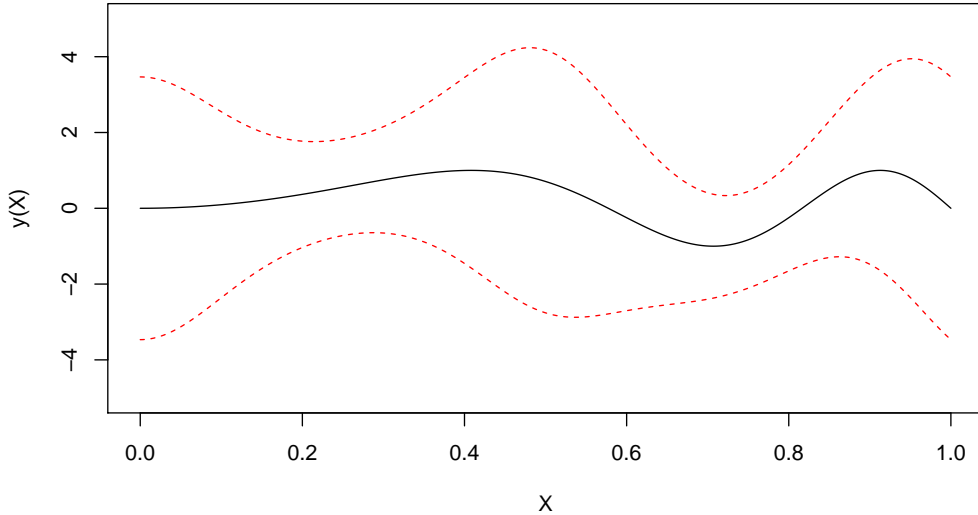


Figure 4: The mean function  $m(x) \pm 2$  standard deviations  $s(x)$  used in simulations comparing SMASH and MFVB. These functions correspond to mean and standard deviation functions (A) in Figure 5 from [Menictas and Wand \(2015\)](#).

2. We generated  $T = 1024$  independent  $(X_t, Y_t)$  pairs, with the  $X_t$ 's (deterministically) equally spaced on  $(0,1)$ , and  $Y_t|X_t$  as above. Performance is measured by MSE evaluated at the 1024  $X_t$ 's for both the mean and the standard deviation.

The first scenario presents some issues for SMASH because the number of data points is not a power of two, nor are the points equally-spaced. To deal with the first issue, following standard ideas in the wavelet literature, we first mirrored the data about the right edge and extract the first  $2^{\lfloor \log_2(2T) \rfloor}$  sample points, so that the number of data points in the new “dataset” is a power of two, and the mean curve is continuous at the right edge of the original data. To further ensure that the input to SMASH is periodic, we reflected the new dataset about its right edge and used this as the final input. To deal with the second issue we follow the common practice of treating the observations as if they were evenly spaced (see [Sardy et al. \(1999\)](#) for discussion). To estimate the original mean and variance functions, we extract the first  $T$  points from the SMASH estimated mean and variance. To evaluate MSE at

	Scenario 1		Scenario 2	
	MSE (for mean)	MSE (for sd)	MSE (for mean)	MSE (for sd)
MFVB	0.0330	0.0199	0.0172	0.0085
SMASH	0.0334	0.0187	0.0158	0.0065

Table 1: Comparison of accuracy (MSE) of SMASH and MFVB for two simulation scenarios. True mean and sd functions are shown in Figure 4. In Scenario 1 the data are not equally spaced and not a power of 2; here SMASH is comparable to MFVB in mean estimation and more accurate for sd estimation. In Scenario 2 the data are equally spaced and a power of 2; here SMASH outperforms MFVB in both mean and sd estimation.

the 201 equally spaced points (Scenario 1) we use simple linear interpolation between the estimated points.

Table 1 shows mean MSEs over 100 independent runs for each scenario. Despite the fact that these simulation scenarios – particularly Scenario 1 – seem better suited to MFVB than SMASH, SMASH performs comparably or better than MFVB for both mean and variance estimation in both Scenarios.

### 3.2.2 Poisson Data

For Poisson data we compared SMASH with six other methods, but we focus here on the comparisons with the best-performing other methods, which are Haar-Fisz (HF) (Fryzlewicz and Nason, 2004) and BMSM (Kolaczyk, 1999). The latter, like SMASH, is an EB method, but with a less flexible prior distribution on the multi-scale coefficients. The HF method involves first performing a transformation on the Poisson counts, and applying Gaussian wavelet methods to the transformed data. There are many choices for Gaussian wavelet methods, and performance depends on these choices, with different choices being optimal for different data sets. The settings we used here for HF are documented in Supplementary Information, and were chosen by us to optimize (average) performance through moderately extensive experimentation on a range of simulations.

To compare the methods we simulated data from several different test functions from Timmermann and Nowak (1999), Fryzlewicz and Nason (2004) and Besbeas et al. (2004). We varied the minimum and maximum intensity of each test function, using (min,max) intensities of (0.01,3), (1/8,8) and (1/128,128). For each test function and intensity level we simulated 100 datasets, each with  $T = 1024$  data points. We focus on results for the first

two intensity settings, which have smaller average intensities. These settings produce smaller average counts, making them more challenging, and also more representative of the kinds of genomic application that we consider below. Complete results are included in Supplementary Materials.

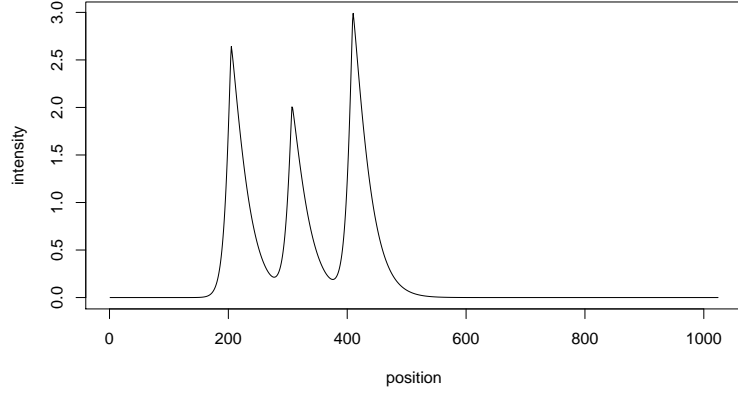
In summary, SMASH outperformed both HF and BMSM in the majority of simulations, with the gain in accuracy being strongest for the more challenging lower-intensity scenarios. A typical result is shown in Figure 5. Among the other two methods, BMSM is the more consistent performer. HF, with the settings used here, performs quite variably, being worse than the other methods in most scenarios, but occasionally performing the best (specifically for the Angles, Bursts and Spikes test functions, with (min,max) intensity (1/128,128)). As noted above, the HF transform can be used with many settings, so results here should be viewed only as a guide to potential performance.

One disadvantage of the HF transform is that, to achieve translation invariance, the transform has to be done explicitly for each shift of the data: the tricks usually used to do this efficiently (Coifman and Donoho, 1995) do not work here. Thus, making HF fully translation invariant increases computation by a factor of  $T$ , rather than the factor of  $\log(T)$  for the other methods. Here we follow advice in Fryzlewicz and Nason (2004) to reduce the computational burden by averaging over 50 shifts of the data rather than  $T$ . Even so, HF was substantially slower than the other methods. A direct comparison of computational efficiency between SMASH and BMSM is difficult, as they are coded in different programming environments. Nevertheless, similarities between the two methods suggest that they should have similar computational cost. Both SMASH and BMSM took, typically, less than a second per dataset in our simulations.

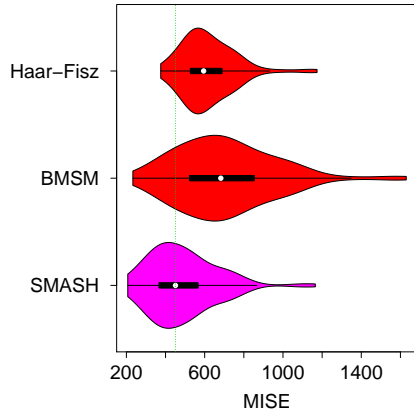
## 4 Applications

### Motorcycle Acceleration Data

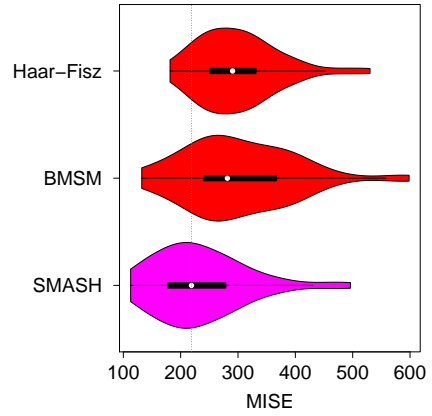
To further illustrate the heteroskedastic Gaussian version of SMASH, we apply it to the motorcycle acceleration dataset from Silverman (1985). The data consist of 133 observations measuring head acceleration in a simulated motorcycle accident that is used to test crash helmets. The dependent variable is *acceleration* (in  $g$ ), and the independent variable is *time* (in  $ms$ ). To



(a)



(b)



(c)

Figure 5: Comparison of methods for denoising Poisson data for the “Bursts” test function. Panel (a) shows the (unscaled) test function. The violin plots in (b) and (c) show distributions of MISE for each method over 100 datasets, with smaller values indicating better performance. Panel (b) is for simulations with a (min,max) intensity of (0.01,3), and panel (c) is for simulations with a (min,max) intensity of (1/8,8). The dashed green line indicates the median MISE for SMASH.

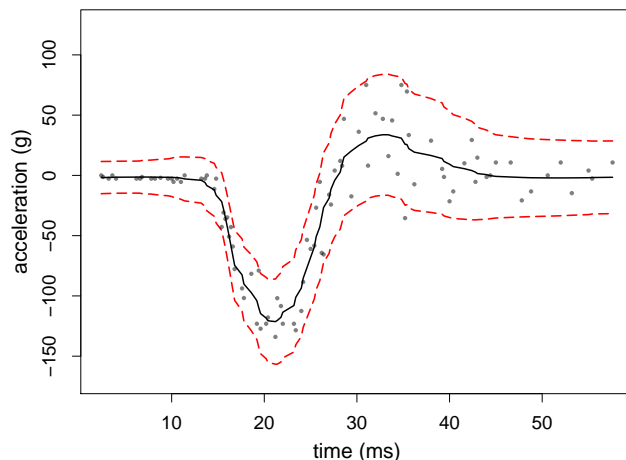


Figure 6: Results from fitting SMASH to the motorcycle acceleration data discussed in [Silverman \(1985\)](#). The figure shows the estimate mean curve (solid black line) with  $\pm 2$  the estimated standard deviation curve (dashed red line).

deal with repeated measurements, we take the median of the measurements for acceleration for any given *time* value. As in Section 3.2.1 we treat the data as if they are equally spaced although they are not. The fitted mean and variance curves (Figure 6) provide a visually appealing fit to the data, and were achieved without hand tuning of any parameters. This contrasts with results in [Delouille et al. \(2004\)](#) – which also uses a wavelet-based approach for heteroskedastic variance, but accounts for the unequal spacing of the data – which required the *ad hoc* removal of high-resolution wavelet coefficients to produce a visual appealing fit.

## ChIP-Seq Data

Chromatin immunoprecipitation sequencing (ChIP-seq) is used to measure transcription factor binding along the genome. After pre-processing (read mapping), the data consist of counts of sequencing reads mapping to each location in genome. These can be treated as arising from an inhomogeneous Poisson process, whose intensity at base  $b$  is related to the strength of the binding of the transcription factor near  $b$ . Because binding tends to be quite localized, the intensity is low on average (the vast majority of counts are 0), but has a small number of intense “peaks”. Identifying these peaks can help

discover regions where binding occurs, which is an important component of understanding gene regulation. Consequently there are many methods published for “peak detection” in ChIP-Seq data (Wilbanks and Facciotti, 2010). Our goal here is to briefly outline how Poisson SMASH could provide an alternative approach to the analysis of ChIP-seq data. The idea is simply to estimate the underlying intensity function, and then identify “peaks” as regions where the estimated intensity exceeds some threshold.

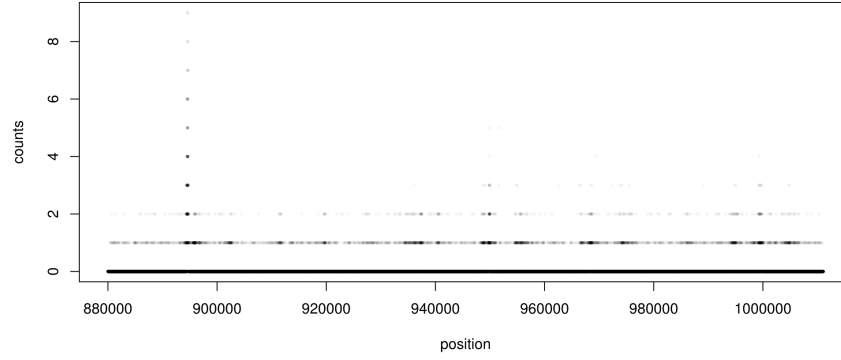
To illustrate, we applied SMASH to a small example ChIP-seq dataset. The resulting estimated intensity is shown in Figure 7b, overlaid on peaks called by the popular peak calling software MACS (Zhang et al., 2008). The locations with the strongest SMASH intensity estimates corresponds to peaks found by MACS. However, the intensity estimate also suggests the presence of several additional weaker peaks not identified by MACS.

The reliable calling of peaks in ChIP-seq data is a multi-faceted problem, and a full assessment lies outside the scope of this paper. Nonetheless, these results suggest that this approach could be worth pursuing. One nice feature of our multi-scale Poisson approach is that it deals well with a range of intensity functions, and could perform well even in settings where peaks are broad and/or not especially well defined. In contrast, the performance of different peak-finding algorithms is often reported to be quite sensitive to the “kinds” of peak that are present, so an algorithm that performs well in one setting may perform poorly in another.

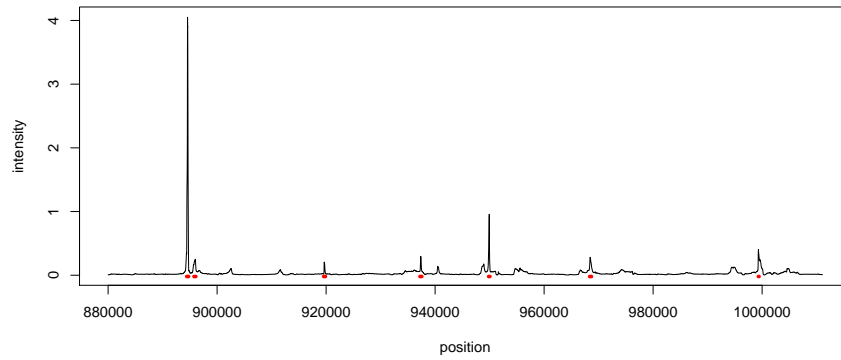
## 5 Discussion

We have demonstrated “smoothing via adaptive shrinkage” (SMASH) for smoothing Gaussian and Poisson data using multi-scale methods. The method is built on the EB shrinkage method, *ash*, whose two key features are i) using a flexible family of unimodal distributions to model the multi-scale coefficients; and ii) to account for varying precision in each coefficient. The first feature allows *ash* to flexibly adapt the amount of shrinkage to the data in hand, so that data that “look smooth” are smoothed more strongly than data that do not. The second feature allows *ash* to deal effectively with heteroskedastic variances, and has the consequence that the mean gets smoothed more strongly in regions where the variance is bigger.

Notably, and unlike many wavelet shrinkage approaches, SMASH is “self-tuning”, and requires no specification of a “primary resolution level” (e.g.



(a)



(b)

Figure 7: Illustration of the potential for SMASH to identify peaks in ChIP-seq data. The data are ChIP-seq for the transcription factor *YY1* in cell line GM12878 collected by the ENCODE (**E**ncyclopedia **O**f **D**NA **E**lements) project, from a region of length  $2^{17} (\approx 131k)$  basepairs from chromosome 1 (hg19 chr1:880001-1011072). Panel (a) shows counts (summed across two replicate experiments). Due to overplotting, darker regions of the plot correspond to higher concentrations of data points. Panel (b) shows the estimated intensity function from SMASH (black solid line) and location of peaks called by MACS (red markers beneath the estimated intensity).

(Nason, 2002) or other tuning parameters. This feature is due to the “adaptive” nature of *ash* noted above: when a particular resolution level shows no strong signal in the data, *ash* learns this and adapts the amount of shrinkage/smoothing appropriately. This self-tuning capability is important for two reasons. First it makes the method easy to use by non-experts, who may find appropriate specification of tuning parameters challenging. Second, it means that the method can be safely applied “in production” to large numbers of datasets in settings, like genomics, where it is impractical to hand-select appropriate tuning parameters separately for every dataset.

Our results here demonstrate that SMASH provides a flexible, fast and accurate approach to smoothing and denoising. We illustrated the flexibility by applying it to two challenging problems - Gaussian heteroskedastic regression, and smoothing of Poisson signals. In both cases our method is consistently competitive in accuracy with existing approaches. And although SMASH requires more computation than a simple thresholding rule, it is fast enough to deal with large problems. This is partly because optimizing over the unimodal distribution in *ash* is a convex optimization problem that can be solved very stably and quickly (Stephens, 2016). For example, our implementation, which exploits the `mosek` convex optimization library (Aps, 2015), typically takes less than 30s for 100,000 observations. SMASH requires multiple applications of *ash* (specifically, because we apply *ash* at each resolution level, it takes  $\log_2(T)$  applications of *ash* in the Poisson case), but is still fast enough to be practical for moderately large problems. For example, smoothing a signal of length 32,768 ( $2^{15}$ ) takes less than a minute for the Poisson case, and less than 2 minutes for the Gaussian, on a modern laptop. It is likely these times could be further improved by more efficient implementations.

Besides its accuracy for point estimation, SMASH also has the advantage that it naturally provides measures of uncertainty in estimated wavelet coefficients, which in turn provide measures of uncertainty (e.g. credible bands) for estimated mean and variance functions, which may be useful in some applications.

Although we have focussed here on applications in one dimension, there is nothing to stop *ash* being similarly applied to multi-scale approaches in higher dimensions, such as image denoising, as in Nowak (1999) for example. For some types of images alternatives to wavelets, such as curvelets (Candes et al., 2000), may produce better results, and extending our work to those settings could be an interesting area for future work. More generally, *ash*



provides a generic approach to shrinkage estimation that could be useful in a range of applications beyond the signal-denoising and smoothing applications considered here.

## 6 Reference

## Appendix A

### Variance estimation for Gaussian denoising

With  $\mathbf{Z}$  as defined in (10), we apply the wavelet transform  $W$  to  $\mathbf{Z}^2$ , and obtain the wavelet coefficients  $\boldsymbol{\delta} = W\mathbf{Z}^2$ . Note that  $\mathbb{E}(\boldsymbol{\delta}) = (\boldsymbol{\gamma})$ , where  $\boldsymbol{\gamma} = W\boldsymbol{\sigma}^2$ . We treat the likelihood for  $\boldsymbol{\gamma}$  as if it were independent, resulting in

$$L(\boldsymbol{\gamma}|\boldsymbol{\delta}) = \prod_{j=0}^J \prod_{k=0}^{T-1} P(\delta_{jk}|\gamma_{jk}). \quad (19)$$

The likelihoods  $L(\gamma_{jk}|\delta_{jk})$  are not normal, but we approximate the likelihood by a normal likelihood through matching the moments of a normal distribution to the distribution  $P(\delta_{jk}|\gamma_{jk})$ . That is,

$$P(\delta_{jk}|\gamma_{jk}) \approx N(\gamma_{jk}, \hat{\mathbb{V}}(\delta_{jk})) \quad (20)$$

so that

$$L(\gamma_{jk}|\delta_{jk}) \approx \phi(\delta_{jk}; \gamma_{jk}, \mathbb{V}(\delta_{jk})) \quad (21)$$

where  $\phi$  is the normal density function, and  $\mathbb{V}(\delta_{jk})$  is the variance of the empirical wavelet coefficients. Since these variances are unknown, we estimate them from the data and then proceed to treat them as known. More specifically, since  $Z_t \sim N(0, \sigma_t^2)$ , we have that

$$\begin{aligned} \mathbb{E}(Z_t^4) &\approx 3\sigma_t^4 \\ \mathbb{V}(Z_t^2) &\approx 2\sigma_t^4 \end{aligned} \quad (22)$$

and so we simply use  $\frac{2}{3}Z_t^4$  as an unbiased estimator for  $\mathbb{V}(Z_t^2)$ . It then follows that  $\hat{\mathbb{V}}(\delta_{jk})$  is given by  $\sum_{l=1}^T \frac{2}{3}Z_l^4 W_{jk,l}^2$ , and is unbiased for  $\mathbb{V}(\delta_{jk})$ .

These will be the inputs to *ash*, which then produces shrunk estimates in the form of posterior means for the corresponding parameters. Although this works well in most cases, there are variance functions for which the above procedure tends to overshrink the wavelet coefficients at the finer levels. This is likely because the distribution of the wavelet coefficients is extremely skewed, especially when the true coefficients are small (at coarser levels the distributions are much less skewed since we are dealing a linear combination of a large number of data points). One way around this issue is to employ a procedure that jointly shrinks the coefficients  $\gamma$  and their variance estimates (implemented in the *jash* option in our software). The final estimate of the variance function is obtained from the posterior means via the average basis inverse across all the shifts.

## Appendix B Poisson denoising

First summarize the data in a recursive manner by defining:

$$Y_{Jk} \equiv Y_k \quad (23)$$

for  $k = 1, \dots, T$ , with  $T = 2^J$ , and

$$Y_{jk} = Y_{j+1,2k} + Y_{j+1,2k+1} \quad (24)$$

for resolution  $j = 0, \dots, J - 1$  and location  $k = 0, \dots, 2^j - 1$ . Hence, we are summing more blocks of observations as we move to coarser levels.

This recursive scheme leads to:

$$Y_{jk} = \sum_{l=k2^{J-j}+1}^{(k+1)2^{J-j}} Y_l \quad (25)$$

for  $j = 0, \dots, J$  and  $k = 0, \dots, 2^j - 1$ .

Similarly, define the following:

$$\mu_{Jk} \equiv \mu_k \quad (26)$$

for  $k = 1, \dots, T$ , and

$$\mu_{jk} = \mu_{j+1,2k} + \mu_{j+1,2k+1} \quad (27)$$

for  $j = 0, \dots, J-1$  and  $k = 0, \dots, 2^j - 1$ . And define

$$\alpha_{jk} = \log(\mu_{j+1,2k}) - \log(\mu_{j+1,2k+1}) \quad (28)$$

$$(29)$$

for  $s = 0, \dots, J-1$  and  $l = 0, \dots, 2^j - 1$ . The  $\alpha$ 's defined this way are analogous to the (true) Haar wavelet coefficients for Gaussian signals.

Using this recursive representation, the likelihood for  $\alpha$  factorizes into a product of likelihoods, where  $\alpha$  is the vector of all the  $\alpha_{sl}$ 's. See [Kolaczyk \(1999\)](#) for example. Specifically,

$$L(\alpha|\mathbf{Y}) = P(\mathbf{Y}|\alpha) \quad (30)$$

$$= P(Y_{0,0}|\mu_{0,0}) \prod_{j=0}^{J-1} \prod_{k=0}^{2^j-1} P(Y_{j+1,2k}|Y_{j,k}, \alpha_{j,k}) \quad (31)$$

$$= L(\mu_{0,0}|Y_{0,0}) \prod_{j=0}^{J-1} \prod_{k=0}^{2^j-1} L(\alpha_{j,k}|Y_{j+1,2k}, Y_{j,k}). \quad (32)$$

Note that  $Y_{00}|\mu_{00} \sim \text{Pois}(\mu_{00})$ . For any given  $j, k$ ,  $Y_{jk}$  is a sum of two independent Poisson random variables, and is itself a Poisson random variable. Hence

$$Y_{j+1,2k}|Y_{jk}, \alpha_{jk} \sim \text{Bin}(Y_{jk}, \frac{1}{1 + e^{-\alpha_{jk}}} \equiv \frac{\mu_{j+1,2k}}{\mu_{jk}})$$

## B.1 Estimates and standard errors for $\alpha_j$

Each  $\alpha_j$  is a ratio of the form  $\log(\mu_{a:b}/\mu_{c:d})$  whose maximum likelihood estimate (mle) is  $\log(Y_{a:b}/Y_{c:d})$ . The main challenge here is that the mle is not well behaved when either the numerator or denominator of  $Y_{a:b}/Y_{c:d}$  is 0. To deal with this, when either is 0 we use Tukey's modification ([Gart and Zweifel, 1967](#)). Specifically, letting  $S$  denote  $Y_{a:b}$ ,  $F$  denote  $Y_{c:d}$  and  $N = S + F$  (corresponding to thinking of these as successes and failures in a binomial experiment, given  $Y_{a:b} + Y_{c:d}$ ), we use

$$\hat{\alpha} = \begin{cases} \log\{(S + 0.5)/(F + 0.5)\} - 0.5 & S = 0 \\ \log\{S/F\} & S = 1, 2, \dots, N-1 \\ \log\{(S + 0.5)/(F + 0.5)\} + 0.5 & S = N \end{cases} \quad (33)$$

$$se(\hat{\alpha}) = \sqrt{V^*(\hat{\alpha}) - \frac{1}{2}\{V_3(\hat{\alpha})\}^2 \left\{V_3(\hat{\alpha}) - \frac{4}{N}\right\}} \quad (34)$$

where

$$V_3(\hat{\alpha}) = \frac{N+1}{N} \left( \frac{1}{S+1} + \frac{1}{F+1} \right) \quad S = 0, \dots, N \quad (35)$$

$$V^*(\hat{\alpha}) = V_3(\hat{\alpha}) \left\{ 1 - \frac{2}{N} + \frac{V_3(\hat{\alpha})}{2} \right\} \quad (36)$$

The square of the standard error in (34) corresponds to  $V^{**}$  from [Gart and Zweifel \(1967, pp. 182\)](#), and is chosen because it is less biased for the true variance of  $\hat{\alpha}$  (when  $N$  is small) as compared to the asymptotic variance of the MLE (see [Gart and Zweifel, 1967](#)). The other two variance estimators from [Gart and Zweifel \(1967\)](#),  $V_1^{++}$  and  $V^{++}$ , were also considered in simulations and gave similar results, but  $V^{**}$  was chosen for its simple form.

## B.2 Signal reconstruction

The first step to reconstructing the signal is to find the posterior means of  $p_{jk} := \frac{\mu_{j+1,2k}}{\mu_{jk}}$  and  $q_{jk} := \frac{\mu_{j+1,2k+1}}{\mu_{jk}}$  (for  $j = 0, \dots, J-1$  and  $k = 0, \dots, 2^j - 1$ ). Specifically, for each  $j$  and  $k$ , we require

$$E(p_{jk}) \equiv E \left( \frac{e^{\alpha_{jk}}}{1 + e^{\alpha_{jk}}} \right) \quad (37)$$

$$E(q_{jk}) \equiv E \left( \frac{e^{-\alpha_{jk}}}{1 + e^{-\alpha_{jk}}} \right). \quad (38)$$

Given the posterior means and variances for  $\alpha_{jk}$  from *ash*, we can approximate (37)-(38) using the Delta method. First, define

$$f(x) = \frac{e^x}{1 + e^x} \quad (39)$$

and consider the Taylor expansion of  $f(x)$  about  $f(E(x))$ :

$$f(x) \approx f(E(x)) + f'(E(x))(x - E(x)) + \frac{f''(E(x))}{2}(x - E(x))^2 \quad (40)$$

where

$$f'(x) = \frac{e^x}{(1 + e^x)^2} \quad (41)$$

$$f''(x) = \frac{e^x(1 - e^x)}{(1 + e^x)^3} \quad (42)$$

Thus

$$E(p_{jk}) \approx f(E(\alpha_{jk})) + \frac{f''(E(\alpha_{jk}))}{2} Var(\alpha_{jk}), \quad (43)$$

$$E(q_{jk}) \approx f(-E(\alpha_{jk})) + \frac{f''(-E(\alpha_{jk}))}{2} Var(\alpha_{jk}), \quad (44)$$

which can be computed by plugging in  $E(\alpha)$  and  $Var(\alpha)$  from *ash*.

Finally, we approximate the posterior mean for  $\mu_t$  by noting that  $\mu_t$  can be written as a product of the  $p$ 's and  $q$ 's for any  $i = 1, 2, \dots, T$ . Specifically, let  $\{c_1, \dots, c_J\}$  be the binary representation of  $i - 1$ , and  $d_m = \sum_{j=1}^m c_j 2^{m-j}$  for  $j = 1, \dots, J - 1$ . Then

$$\mu_k = \mu_{00} p_{00}^{1-c_1} p_{1,d_1}^{1-c_2} \dots p_{J-1,d_{J-1}}^{1-c_J} q_{00}^{c_1} q_{1,d_1}^{c_2} \dots q_{J-1,d_{J-1}}^{c_J} \quad (45)$$

where we usually estimate  $\mu_{00}$  by  $\sum_l Y_l$  (see Kolaczyk (1999)). Using the independence of the  $p$ 's and  $q$ 's from different scales, we have:

$$E(\mu_t) = \mu_{00} E(p_{00})^{1-c_1} E(p_{1,d_1})^{1-c_2} \dots E(p_{J-1,d_{J-1}})^{1-c_J} \\ \cdot E(q_{00})^{c_1} E(q_{1,d_1})^{c_2} \dots E(q_{J-1,d_{J-1}})^{c_J}. \quad (46)$$

We can also approximate the posterior variance of  $\mu_t$ . (This allows creation of an approximate credible interval by making a normal approximation.) From (45) we have:

$$E(\mu_t^2) = \mu_{00}^2 E(p_{00}^2)^{1-c_1} E(p_{1,d_1}^2)^{1-c_2} \dots E(p_{J-1,d_{J-1}}^2)^{1-c_J} \\ \cdot E(q_{00}^2)^{c_1} E(q_{1,d_1}^2)^{c_2} \dots E(q_{J-1,d_{J-1}}^2)^{c_J}. \quad (47)$$

To compute this we again use the Delta method (with  $f(x) = (\frac{e^x}{1+e^x})^2$ ) to obtain:

$$E(p_{jk}^2) \approx \left( f(E(\alpha_{jk})) + \frac{f''(E(\alpha_{jk}))}{2} Var(\alpha_{jk}) \right)^2 + \\ \{f'(E(\alpha_{jk}))\}^2 Var(\alpha_{jk}) \quad (48)$$

$$E(q_{jk}^2) \approx \left( f(-E(\alpha_{jk})) + \frac{f''(-E(\alpha_{jk}))}{2} Var(\alpha_{jk}) \right)^2 + \\ \{f'(-E(\alpha_{jk}))\}^2 Var(\alpha_{jk}). \quad (49)$$

Finally we combine (46) and (47) to find  $Var(\mu_k)$ .

### B.3 Translation Invariance

It is common in multi-scale analysis to perform analyses over all  $T$  circulant shifts of the data, because this is known to consistently improve accuracy. (The  $t$ -th circulant shift of the signal  $\mathbf{Y}$  is created from  $\mathbf{Y}$  by moving the first  $T - t$  elements of  $\mathbf{Y}$   $t$  positions to the right and then putting the last  $t$  elements of  $\mathbf{Y}$  in the first  $t$  locations.)

To implement this in practice, we begin by computing the  $\alpha$  coefficients, and their corresponding standard errors, for all  $T$  circulant shifts of the data. This is done efficiently (in  $O(\log_2 T)$  operations), using ideas from [Coifman and Donoho \(1995\)](#). Details are as in [Kolaczyk \(1999\)](#); indeed our software implementation benefited from Matlab code provided by [Kolaczyk \(1999\)](#) for the TI table construction, which we ported to C++ and integrated with R using Rcpp ([Eddelbuettel et al., 2011](#)).

This yields a table of  $\alpha$  coefficients, with  $T$  coefficients at each of  $\log_2(T)$  resolution levels, and a corresponding table of standard errors. As in the Gaussian case we then apply *ash* separately to the  $T$  coefficients at each resolution level, to obtain a posterior mean and posterior variance for each  $\alpha$ . We then use the methods above to compute quantities of interest averaged over all  $T$  shifts of the data. For example, our final estimate of the mean signal is given by

$$\frac{1}{T} \sum_{t=1}^T \hat{\mu}_k^{(t)} \quad (50)$$

where  $\hat{\mu}_k^{(t)}$  denotes the posterior mean of  $\mu_k$  computed from the  $t$ -th circulant shift of the data. Again, using idea from [Coifman and Donoho \(1995\)](#) this averaging can be done in  $O(\log_2 T)$  operations.

## References

- Antoniadis, A., J. Bigot, and T. Sapatinas (2001). Wavelet Estimators in Nonparametric Regression: A Comparative Simulation Study. *Journal of Statistical Software* 6(6). [4](#), [13](#)
- Aps, M. (2015). *Rmosek: The R to MOSEK Optimization Interface*. R package version 7.1.2. [24](#)

- Besbeas, P., I. De Feis, and T. Sapatinas (2004, August). A Comparative Simulation Study of Wavelet Shrinkage Estimators for Poisson Counts. *International Statistical Review* 72(2), 209–237. [4](#), [18](#)
- Beylkin, G. (1992). On the Representation of Operators in Bases of Compactly Supported Wavelets. *SIAM Journal on Numerical Analysis* 29(6), 1716–1740. [8](#)
- Bickel, P. J. and E. Levina (2008, December). Covariance regularization by thresholding. *The Annals of Statistics* 36(6), 2577–2604. [3](#)
- Brown, L. D. and M. Levine (2007, October). Variance estimation in non-parametric regression via the difference sequence method. *The Annals of Statistics* 35(5), 2219–2232. [15](#)
- Cai, T. T. and L. Wang (2008, October). Adaptive variance function estimation in heteroscedastic nonparametric regression. *The Annals of Statistics* 36(5), 2025–2054. [8](#), [15](#)
- Candes, E. J., J. C. Emmanuel, and D. L. Donoho (2000). Curvelets - A Surprisingly Effective Nonadaptive Representation For Objects with Edges. [24](#)
- Clyde, M. and E. I. George (2000, January). Flexible Empirical Bayes estimation for wavelets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 62(4), 681–698. [3](#), [5](#)
- Coifman, R. R. and D. L. Donoho (1995). Translation-invariant De-noising. [3](#), [7](#), [8](#), [13](#), [19](#), [30](#)
- Daniels, M. J. and R. E. Kass (2001). Shrinkage Estimators for Covariance Matrices. *Biometrics* 57(4), 1173–1184. [3](#)
- Daubechies, I. (1992). *Ten lectures on wavelets*, Volume 61 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM). [13](#)
- Delouille, V., J. Simoens, and R. von Sachs (2004, September). Smooth Design-Adapted Wavelets for Nonparametric Stochastic Regression. *Journal of the American Statistical Association* 99(467), 643–658. [8](#), [21](#)

- Donoho, D. L. and I. M. Johnstone (1995, December). Adapting to Unknown Smoothness via Wavelet Shrinkage. *Journal of the American Statistical Association* 90(432), 1200–1224. [3](#)
- Donoho, D. L. and J. M. Johnstone (1994, September). Ideal spatial adaptation by wavelet shrinkage. *Biometrika* 81(3), 425–455. [3](#)
- Eddelbuettel, D., R. François, J. Allaire, J. Chambers, D. Bates, and K. Ushey (2011). Rcpp: Seamless R and C++ integration. *Journal of Statistical Software* 40(8), 1–18. [30](#)
- Efron, B. (2004). Large-Scale Simultaneous Hypothesis Testing: The Choice of a Null Hypothesis. *Journal of the American Statistical Association* 99(465), 96–104. [3](#)
- Efron, B. and R. Tibshirani (2002, June). Empirical bayes methods and false discovery rates for microarrays. *Genet. Epidemiol.* 23(1), 70–86. [3](#)
- Fan, J. and Q. Yao (1998). Efficient Estimation of Conditional Variance Functions in Stochastic Regression. *Biometrika* 85(3), 645–660. [15](#)
- Fryzlewicz, P. and G. P. Nason (2004, September). A Haar-Fisz Algorithm for Poisson Intensity Estimation. *Journal of Computational and Graphical Statistics* 13(3), 621–638. [18](#), [19](#)
- Gao, H.-y. (1997). Wavelet Shrinkage Estimates For Heteroscedastic Regression Models. [11](#), [13](#)
- Gart, J. J. and J. R. Zweifel (1967, June). On the bias of various estimators of the logit and its variance with application to quantal bioassay. *Biometrika* 54(1), 181–187. [10](#), [27](#), [28](#)
- Johnstone, I. M. and B. W. Silverman (2005, August). Empirical Bayes selection of wavelet thresholds. *The Annals of Statistics* 33(4), 1700–1752. [3](#), [5](#), [7](#), [13](#)
- Koenker, R. and I. Mizera (2014). Convex optimization, shape constraints, compound decisions, and Empirical Bayes rules. *Journal of the American Statistical Association* 109(506), 674–685. [5](#)



- Kolaczyk, E. D. (1999, September). Bayesian Multiscale Models for Poisson Processes. *Journal of the American Statistical Association* 94(447), 920–933. [4](#), [9](#), [10](#), [18](#), [27](#), [30](#)
- Menictas, M. and M. P. Wand (2015, March). Variational Inference for Heteroscedastic Semiparametric Regression. *Aust. N. Z. J. Stat.* 57(1), 119–138. [15](#), [17](#)
- Nason, G. (2002). Choice of wavelet smoothness, primary resolution and threshold in wavelet shrinkage. *12*(3), 219–227. [24](#)
- Nason, G. (2013). *wavethresh: Wavelets statistics and transforms*. R package version 4.6.6. [8](#)
- Nowak, R. D. (1999). Multiscale Hidden Markov Models for Bayesian Image Analysis. In P. Müller and B. Vidakovic (Eds.), *Bayesian Inference in Wavelet-Based Models*, Volume 141 of *Lecture Notes in Statistics*, pp. 243–265. Springer New York. [24](#)
- Nowak, R. D. and E. D. Kolaczyk (2000, August). A statistical multiscale framework for Poisson inverse problems. *Information Theory, IEEE Transactions on* 46(5), 1811–1825. [9](#)
- Sardy, S., D. B. Percival, A. G. Bruce, H.-Y. Gao, and W. Stuetzle (1999, April). Wavelet shrinkage for unequally spaced data. *Statistics and Computing* 9(1), 65–75. [17](#)
- Silverman, B. W. (1985). Some Aspects of the Spline Smoothing Approach to Non-Parametric Regression Curve Fitting. *Journal of the Royal Statistical Society. Series B (Methodological)* 47(1), 1–52. [19](#), [21](#)
- Silverman, B. W. (1999). Wavelets in Statistics: Beyond the Standard Assumptions. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences* 357(1760), 2459–2473. [7](#)
- Stephens, M. (2016). False Discovery Rates: A New Deal. *bioRxiv*. [4](#), [5](#), [24](#)
- Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 58(1), 267–288. [3](#)

- Timmermann, K. E. and R. D. Nowak (1999, April). Multiscale modeling and estimation of Poisson processes with application to photon-limited imaging. *Information Theory, IEEE Transactions on* 45(3), 846–862. [9](#), [10](#), [18](#)
- Varadhan, R. (2014). *SQUAREM: Squared extrapolation methods for accelerating fixed-point iterations*. R package version 2014.8-1. [5](#)
- Wilbanks, E. G. and M. T. Facciotti (2010, July). Evaluation of Algorithm Performance in ChIP-Seq Peak Detection. *PLoS ONE* 5(7), e11471+. [22](#)
- Zhang, Y., T. Liu, C. A. Meyer, J. Eeckhoute, D. S. Johnson, B. E. Bernstein, C. Nusbaum, R. M. Myers, M. Brown, W. Li, and X. S. Liu (2008, September). Model-based analysis of ChIP-Seq (MACS). *Genome biology* 9(9), R137+. [22](#)